Acceleration for Compressed Gradient Descent in Distributed and Federated Optimization

Zhize Li [1] **Dmitry Kovalev** [1] **Xun Qian** [1] **Peter Richtárik** [1]

Abstract

Due to the high communication cost in distributed and federated learning problems, methods relying on compression of communicated messages are becoming increasingly popular. While in other contexts the best performing gradient-type methods invariably rely on some form of acceleration/momentum to reduce the number of iterations, there are no methods which combine the benefits of both gradient compression and acceleration. In this paper, we remedy this situation and propose the first *accelerated compressed gradient descent (ACGD)* methods. In the single machine regime, we prove that ACGD enjoys the rate $O\Big((1+\omega)\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\Big)$ for μ-strongly convex problems and $O\Big((1+\omega)\sqrt{\frac{L}{\epsilon}}\Big)$ for convex problems, respectively, where ω is the compression parameter. Our results improve upon the existing non-accelerated rates $O\Big((1+\omega)\frac{L}{\mu}\log\frac{1}{\epsilon}\Big)$ and $O\Big((1+\omega)\frac{L}{\epsilon}\Big)$, respectively, and recover the optimal rates of accelerated gradient descent as a special case when no compression ($\omega=0$) is applied. We further propose a distributed variant of ACGD (called ADIANA) and prove the convergence rate $\widetilde{O}\Big(\omega+\sqrt{\frac{L}{\mu}}+\sqrt{\big(\frac{\omega}{n}+\sqrt{\frac{\omega}{n}}\big)\frac{\omega L}{\mu}}\Big)$, where n is the number of devices/workers and $\widetilde{O}$ hides the logarithmic factor $\log\frac{1}{\epsilon}$. This improves upon the previous best result $\widetilde{O}\Big(\omega+\frac{L}{\mu}+\frac{\omega L}{n\mu}\Big)$ achieved by the DIANA method of Mishchenko et al. (2019). Finally, we conduct several experiments on real-world datasets which corroborate our theoretical results and confirm the practical superiority of our accelerated methods.

[1]King Abdullah University of Science and Technology, Thuwal, Kingdom of Saudi Arabia. Correspondence to: Zhize Li <zhize.li@kaust.edu.sa>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

1. Introduction

With the proliferation of edge devices such as mobile phones, wearables and smart home devices comes an increase in the amount of data rich in potential information which can be mined for the benefit of the users. One of the approaches of turning the raw data into information is via federated learning (Konečný et al., 2016; McMahan et al., 2017), where typically a single global supervised model is trained in a massively distributed manner over a network of heterogeneous devices.

Training supervised federated learning models is typically performed by solving an optimization problem of the form

$$\min_{x\in\mathbb{R}^d}\Big\{P(x):=\frac{1}{n}\sum_{i=1}^{n}f_i(x)+\psi(x)\Big\}, \tag{1}$$

where $f_i:\mathbb{R}^d\to\mathbb{R}$ is smooth loss associated with data stored on device i and $\psi:\mathbb{R}^d\to\mathbb{R}\cup\{+\infty\}$ is a relatively simple but possibly nonsmooth regularizer.

In distributed learning in general, and federated learning in particular, communication of messages across a network forms the bottleneck of the training system. It is thus very important to devise novel strategies for reducing the number of communication rounds. Two of the most common strategies are i) local computations (Ma et al., 2017; Stich, 2019; Khaled et al., 2020) and ii) communication compression (Seide et al., 2014; Alistarh et al., 2017; Wangni et al., 2018; Horváth et al., 2019a). The former is used to perform more local computations on each device before communication and subsequent model averaging, hoping that this will reduce the total number of communications. The latter is used to reduce the size of communicated messages, saving precious time spent in each communication round, and hoping that this will not increase the total number of communications.

1.1. Theoretical inefficiency of local methods

Despite their practical success, local methods are poorly understood and there is much to be discovered. For instance, there exist no theoretical results which would suggest that any local method (e.g., local gradient descent (GD) or local SGD) can achieve better communication complexity than its standard non-local variant (e.g., GD, SGD). In fact, until recently, no complexity results existed for local SGD in environments with heterogeneous data (Khaled et al., 2019; 2020), a key regime in federated learning settings (Li et al., 2019). In the important regime when all participating devices compute full gradients based on their local data, the recently proposed stochastic controlled averaging (SCAFFOLD) method (Karimireddy et al., 2019) offers no improvement on the number of communication as the number of local steps grows despite the fact that this is a rather elaborate method combining local stochastic gradient descent with control variates for reducing the model drift among clients.

1.2. Methods with compressed communication

However, the situation is much brighter with methods employing communication compression. Indeed, several recent theoretical results suggest that by combining an appropriate (typically randomized) compression operator with a suitably designed gradient-type method, one can obtain improvement in the the total communication complexity over comparable baselines not performing any compression. For instance, this is the case for distributed compressed gradient descent (CGD) (Alistarh et al., 2017; Khirirat et al., 2018; Horváth et al., 2019a; Li & Richtárik, 2020) and distributed CGD methods which employ variance reduction to tame the variance introduced by compression (Hanzely et al., 2018; Mishchenko et al., 2019; Horváth et al., 2019b; Hanzely & Richtárik, 2019b; Li & Richtárik, 2020).

While in the case of CGD compression leads to a decrease in the size of communicated messages per communication round, it leads to an increase in the number of communications. Yet, certain compression operators, such as natural dithering (Horváth et al., 2019a), were shown to be better than no compression in terms of the overall communication complexity.

The variance-reduced CGD method DIANA (Mishchenko et al., 2019; Horváth et al., 2019b) enjoys even better behavior: the number of communication rounds for this method is unaffected up to a certain level of compression when the variance induced by compression is smaller than a certain threshold. This threshold can be very large in practice, which means that massive reduction is often possible in the number of communicated bits without this having any adverse effect on the number of communication rounds.

Recall that a function $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth or has L-Lipschitz continuous gradient (for $L > 0$) if

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \tag{2}$$

and μ-strongly convex (for $\mu \geq 0$) if

$$f(x) - f(y) - \langle \nabla f(y), x - y\rangle \geq \frac{\mu}{2}\|x - y\|^2 \tag{3}$$

for all $x, y \in \mathbb{R}^d$. The $\mu = 0$ case corresponds to the standard convexity.

In particular, for L-smooth and μ-strongly convex f with n machines, DIANA enjoys the iteration bound $O\left(\left(\omega + \frac{L}{\mu} + \frac{\omega}{n}\frac{L}{\mu}\right)\log\frac{1}{\epsilon}\right)$, where $\frac{L}{\mu}$ is the condition number and ω is the compression parameter (see Definition 1). If $\omega = 0$, which corresponds to no compression, DIANA recovers the $\mathcal{O}\left(\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$ rate of gradient descent. On the other hand, as long as $\omega = O\left(\min\left\{\frac{L}{\mu}, n\right\}\right)$, the rate is still $\mathcal{O}\left(\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$, which shows that DIANA is able to retain the same number of communication rounds as gradient descent and yet save on bit transmission in each round. The higher ω is allowed to be, the more compression can be applied.

2. Contributions

Discouraged by the lack of theoretical results suggesting that local methods indeed help to reduce the number of communications, and encouraged by the theoretical success of CGD methods, in this paper we seek to enhance CGD methods with a mechanism which, unlike local updating, can provably lead to a decrease of the number of communication rounds.

What mechanism could achieve further improvements?

In the world of deterministic gradient methods, one technique for such a reduction is well known: *Nesterov acceleration / momentum* (Nesterov, 1983; 2004). In case of stochastic gradient methods, the accelerated method Katyusha (Allen-Zhu, 2017) achieves the optimal rate for strongly convex problems, and the unified accelerated method Varag (Lan et al., 2019) achieves the optimal rates for convex problems regardless of the strong convexity. See also (Kovalev et al., 2020; Qian et al., 2019) for some enhancements. Essentially all state-of-the-art methods for training deep learning models, including Adam (Kingma & Ba, 2014), rely on the use of momentum/acceleration in one form or another, albeit lacking in theoretical support.

However, the successful combination of gradient compression and acceleration/momentum has so far remained elusive, and to the best of our knowledge, no algorithms supported with theoretical results exist in this space. Given the omnipresence of momentum in modern machine learning, this is surprising.

We now summarize our key contributions:

2.1. First combination of gradient compression and acceleration

We develop the first gradient-type optimization methods provably combining the benefits of gradient compression

Table 1. Convergence results for the special case with $n = 1$ device (i.e., problem (4))

Algorithm	μ-strongly convex f	convex f
Compressed Gradient Descent (CGD (Khirirat et al., 2018))	$O\left((1+\omega)\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$	$O\left((1+\omega)\frac{L}{\epsilon}\right)$
ACGD (this paper)	$O\left((1+\omega)\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$	$O\left((1+\omega)\sqrt{\frac{L}{\epsilon}}\right)$

Table 2. Convergence results for the general case with n devices (i.e., problem (1)). Our results are always better than previous results.

Algorithm	$n \le \omega$ (few devices or high compression)	$n > \omega$ (lots of devices or low compression)
Distributed CGD (DIANA (Mishchenko et al., 2019))	$O\left(\omega(1+\frac{L}{n\mu})\log\frac{1}{\epsilon}\right)$	$O\left((\omega+\frac{L}{\mu})\log\frac{1}{\epsilon}\right)$
ADIANA (this paper)	$O\left(\omega\left(1+\sqrt{\frac{L}{n\mu}}\right)\log\frac{1}{\epsilon}\right)$	$O\left(\left(\omega+\sqrt{\frac{L}{\mu}}+\sqrt{\sqrt{\frac{\omega}{n}}\frac{\omega L}{\mu}}\right)\log\frac{1}{\epsilon}\right)$

and acceleration: i) ACGD (Algorithm 1) in the single device case, and ii) ADIANA (Algorithm 2) in the distributed case.

2.2. Single device setting

We first study the single-device setting, and design an accelerated CGD method (ACGD - Algorithm 1) for solving the unconstrained smooth minimization problem

$$\min_{x\in\mathbb{R}^d} f(x) \tag{4}$$

in the regimes when f is L-smooth and i) μ-strongly convex, and ii) convex. Our theoretical results are summarized in Table 1. In the strongly convex case, we improve the complexity of CGD (Khirirat et al., 2018) from $O\left((1+\omega)\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$ to $O\left((1+\omega)\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$. In the convex case, the improvement is from $O\left((1+\omega)\frac{L}{\epsilon}\right)$ to $O\left((1+\omega)\sqrt{\frac{L}{\epsilon}}\right)$, where $\omega \ge 0$ is the compression parameter (see Definition 1).

2.3. Distributed setting

We further study the distributed setting with n devices/nodes and focus on problem (1) in its full generality, i.e.,

$$\min_{x\in\mathbb{R}^d}\left\{P(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x)\right\},$$

The presence of multiple nodes ($n > 1$) and of the regularizer ψ poses additional challenges. In order to address them, we need to not only combine acceleration and compression, but also introduce a DIANA-like variance reduction mechanism to remove the variance introduced by the compression operators.

In particular, we have developed an accelerated variant of the DIANA method for solving the general problem (1), which we call ADIANA (Algorithm 2). The comparison of complexity results between ADIANA and DIANA is summarized in Table 2.

Note that our results always improve upon the non-accelerated DIANA method. Indeed, in the regime when the compression parameter ω is larger than the number of nodes n, we improve the DIANA rate $O\left(\omega(1+\frac{L}{n\mu})\log\frac{1}{\epsilon}\right)$ to $O\left(\omega\left(1+\sqrt{\frac{L}{n\mu}}\right)\log\frac{1}{\epsilon}\right)$. On the other hand, in the regime when $\omega < n$, we improve the DIANA rate $O\left((\omega+\frac{L}{\mu})\log\frac{1}{\epsilon}\right)$ to $O\left(\left(\omega+\sqrt{\frac{L}{\mu}}+\sqrt{\sqrt{\frac{\omega}{n}}\frac{\omega L}{\mu}}\right)\log\frac{1}{\epsilon}\right)$. Our rate is better since $\omega+\frac{L}{\mu} \ge 2\sqrt{\frac{\omega L}{\mu}}$ and $\sqrt{\frac{\omega}{n}} < 1$ (note that $\omega < n$).

Note that if $\omega \le n^{1/3}$, which is more often true in federated learning as the number of devices in federated learning is typically very large, our ADIANA result reduces to $O\left(\left(\omega+\sqrt{\frac{L}{\mu}}\right)\log\frac{1}{\epsilon}\right)$. In particular, if $\omega = O\left(\min\left\{n^{1/3},\sqrt{\frac{L}{\mu}}\right\}\right)$, then the communication round is $O\left(\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$; the same as that of non-compressed accelerated gradient descent (AGD) (Nesterov, 2004). It means that ADIANA benefits from cheaper communication due to compression *for free* without hurting the convergence rate (i.e., the communication rounds are the same), and is therefore better suited for federated optimization.

3. Randomized Compression Operators

We now introduce the notion of a randomized compression operator which is used to compress the gradients.

Definition 1 (Compression operator) *A randomized map $\mathcal{C}:\mathbb{R}^d \mapsto \mathbb{R}^d$ is an ω-compression operator if*

$$\mathbb{E}[\mathcal{C}(x)] = x,\ \ \mathbb{E}[\|\mathcal{C}(x)-x\|^2] \le \omega\|x\|^2,\ \ \forall x\in\mathbb{R}^d. \tag{5}$$

In particular, no compression ($\mathcal{C}(x) \equiv x$) implies $\omega = 0$.

Note that the conditions (5) require the compression operator to be unbiased and its variance uniformly bounded by a relative magnitude of the vector which we are compressing.

3.1. Examples

We now give a few examples of randomized compression operators without attempting to be exhaustive.

Example 1 (Random sparsification): Given $x \in \mathbb{R}^d$, the random-k sparsification operator is defined by

$$\mathcal{C}(x) := \frac{d}{k}(\xi_k \odot x),$$

where $\odot$ denotes the Hadamard (element-wise) product and $\xi_k \in \{0,1\}^d$ is a uniformly random binary vector with k nonzero entries ($\|\xi_k\|_0 = k$). This random-k sparsification operator $\mathcal{C}$ satisfies (5) with $\omega = \frac{d}{k} - 1$. Indeed, no compression $k = d$ implies $\omega = 0$.

Example 2 (Quantization): Given $x \in \mathbb{R}^d$, the (p,s)-quantization operator is defined by

$$\mathcal{C}(x) := \text{sign}(x) \cdot \|x\|_p \cdot \frac{1}{s} \cdot \xi_s,$$

where $\xi_s \in \mathbb{R}^d$ is a random vector with i-th element

$$\xi_s(i) := \begin{cases} l+1, & \text{with probability } \frac{|x_i|}{\|x\|_p}s - l \\ l, & \text{otherwise} \end{cases},$$

where the level l satisfies $\frac{|x_i|}{\|x\|_p} \in [\frac{l}{s}, \frac{l+1}{s}]$. The probability is chosen so that $\mathbb{E}[\xi_s(i)] = \frac{|x_i|}{\|x\|_p}s$. This (p,s)-quantization operator $\mathcal{C}$ satisfies (5) with $\omega = 2 + \frac{d^{1/p}+d^{1/2}}{s}$. In particular, QSGD (Alistarh et al., 2017) used $p = 2$ (i.e., $(2,s)$-quantization) and proved that the expected sparsity of $\mathcal{C}(x)$ is $\mathbb{E}[\|\mathcal{C}(x)\|_0] = O\big(s(s+\sqrt{d})\big)$.

4. Accelerated CGD: Single Machine

In this section, we study the special case of problem (1) with a single machine ($n = 1$) and no regularizer ($\psi(x) \equiv 0$), i.e., problem (4):

$$\min_{x \in \mathbb{R}^d} f(x).$$

4.1. The CGD algorithm

First, we recall the update step in compressed gradient descent (CGD) method, i.e.,

$$x^{k+1} = x^k - \eta \mathcal{C}(\nabla f(x^k)),$$

where $\mathcal{C}$ is a kind of ω-compression operator defined in Definition 1.

As mentioned earlier, convergence results of CGD are $O\left((1+\omega)\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$ for strongly convex problems and $O\left((1+\omega)\frac{L}{\epsilon}\right)$ for convex problems (see Table 1). The convergence proof for strongly convex problems, i.e., $O\left((1+\omega)\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$, can be found in (Khirirat et al., 2018). For completeness, we now establish a convergence result for convex problems, i.e., $O\left((1+\omega)\frac{L}{\epsilon}\right)$ since we did not find it in the literature.

Theorem 1 *Suppose f is convex with L-Lipschitz continuous gradient and the compression operator $\mathcal{C}$ satisfies (5). Fixing the step size $\eta = \frac{1}{(1+\omega)L}$, the number of iterations performed by CGD to find an ϵ-solution such that*

$$\mathbb{E}[f(x^k)] - f(x^*) \le \epsilon$$

is at most

$$k = O\left(\frac{(1+\omega)L}{\epsilon}\right).$$

4.2. The ACGD algorithm

Note that in the non-compressed case $\omega = 0$ (i.e., CGD is reduced to standard GD), there exists methods for obtaining accelerated convergence rates of $O\left(\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$ and $O\left(\sqrt{\frac{L}{\epsilon}}\right)$ for strongly convex and convex problems, respectively. However, no accelerated convergence results exist for CGD methods. Inspired by Nesterov's accelerated gradient descent (AGD) method (Nesterov, 2004) and FISTA (Beck & Teboulle, 2009), we propose the first accelerated compressed gradient descent (ACGD) method, described in Algorithm 1.

Algorithm 1 Accelerated CGD (ACGD)

```
Input: initial point $x^0$, $\{\eta_k\}, \{\theta_k\}, \{\beta_k\}, \{\gamma_k\}, p$
1: $z^0 = y^0 = x^0$
2: for $k = 0, 1, 2, \ldots$ do
3:     $x^k = \theta_k y^k + (1-\theta_k) z^k$
4:     Compress gradient $g^k = \mathcal{C}(\nabla f(x^k))$
5:     $y^{k+1} = x^k - \frac{\eta_k}{p} g^k$
6:     $z^{k+1} = \frac{1}{\gamma_k} y^{k+1} + \left(\frac{1}{p} - \frac{1}{\gamma_k}\right) y^k$
           $+ \left(1 - \frac{1}{p}\right)(1-\beta_k) z^k + \left(1 - \frac{1}{p}\right)\beta_k x^k$
7: end for
```

4.3. Convergence theory

Our accelerated convergence results for ACGD (Algorithm 1) are stated in Theorems 2 and 3, formulated next.

Theorem 2 (ACGD: convex case) *Let f be convex with L-Lipschitz continuous gradient and let the compression* *operator $\mathcal{C}$ satisfy (5). Choose the parameters in ACGD (Algorithm 1) as follows:*

$$\eta_k \equiv \frac{1}{L}, \quad p = 1+\omega,$$
$$\theta_k = \frac{k}{k+2}, \quad \beta_k \equiv 0, \quad \gamma_k = \frac{2p}{k+2}.$$

Then the number of iterations performed by ACGD to find an ϵ-solution such that

$$\mathbb{E}[f(x^k)] - f(x^*) \le \epsilon$$

is at most

$$k = O\left((1+\omega)\sqrt{\frac{L}{\epsilon}}\right).$$

Theorem 3 (ACGD: strongly convex case) *Let f be μ-strongly convex with L-Lipschitz continuous gradient and let the compression operator $\mathcal{C}$ satisfy (5).Choose the parameters in ACGD (Algorithm 1) as follows:*

$$\eta_k \equiv \frac{1}{L}, \quad p = 1+\omega,$$
$$\theta_k \equiv \frac{p}{p+\sqrt{\mu/L}}, \quad \beta_k \equiv \frac{\sqrt{\mu/L}}{p}, \quad \gamma_k \equiv \sqrt{\frac{\mu}{L}}.$$

Then the number of iterations performed by ACGD to find an ϵ-solution such that

$$\mathbb{E}[f(x^k)] - f(x^*) \le \epsilon$$

(or $\mathbb{E}[\|x^k - x^*\|^2] \le \epsilon$) is at most

$$k = O\left((1+\omega)\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right).$$

In the non-compressed case $\omega = 0$ (i.e., $\mathcal{C}(x) \equiv x$), our results recover the standard optimal rates of accelerated gradient descent. Further, if we consider the random-k sparsification compression operator, ACGD can be seen as a variant of accelerated randomized coordinate descent (Nesterov, 2012). Our results recover the optimal results of accelerated randomized coordinate descent method (Allen-Zhu et al., 2016; Hanzely & Richtárik, 2019a) under the same standard smoothness assumptions.

4.4. Proof sketch

The following lemma which demonstrates improvement in one iteration plays a key role in our analysis.

Lemma 1 *If parameters $\{\eta_k\}, \{\theta_k\}, \{\beta_k\}, \{\gamma_k\}$ and p satisfy $\theta_k = \frac{1-\gamma_k/p}{1-\beta_k\gamma_k/p}$, $\beta_k \le \min\{\frac{\mu\eta_k}{\gamma_k p}, 1\}$, $p \ge \frac{(1+L\eta_k)(1+\omega)}{2}$ and the compression operator $\mathcal{C}^k$ satisfies (5), then we have for any iteration k of ACGD, and for all $x \in \mathbb{R}^d$,*

$$\frac{2\eta_k}{\gamma_k^2}\mathbb{E}[f(y^{k+1}) - f(x)] + \mathbb{E}[\|z^{k+1} - x\|^2]$$
$$\le \left(1 - \frac{\gamma_k}{p}\right)\frac{2\eta_k}{\gamma_k^2}\left(f(y^k) - f(x)\right) + (1-\beta_k)\|z^k - x\|^2,$$

where the expectation is with respect to the randomness of compression operator sampled at iteration k.

The proof of Theorems 2 and 3 can be derived (i.e., plug into the specified parameters ($\{\eta_k\}, \{\theta_k\}, \{\beta_k\}, \{\gamma_k\}$ and p) and collect all iterations) from Lemma 1. The detailed proofs can be found in the appendix.

5. Accelerated CGD: Distributed Setting

We now turn our attention to the general distributed case, i.e., problem (1):

$$\min_{x\in\mathbb{R}^d}\left\{P(x) := \frac{1}{n}\sum_{i=1}^n f_i(x) + \psi(x)\right\}.$$

The presence of multiple nodes ($n > 1$) and of the regularizer ψ poses additional challenges.

5.1. The ADIANA algorithm

We now propose an *accelerated* algorithm for solving problem (1). Our method combines both acceleration and variance reduction, and hence can be seen as an accelerated version of DIANA (Mishchenko et al., 2019; Horváth et al., 2019b). Therefore, we call our method ADIANA (Algorithm 2). In this case, each machine/agent computes its local gradient (e.g., $\nabla f_i(x^k)$) and a shifted version thereof is compressed and sent to the server. The server subsequently aggregates all received messages, to form a stochastic gradient estimator g^k of $\frac{1}{n}\sum_i \nabla f_i(x^k)$, and then performs a proximal step. The shift terms h_i^k are adaptively changing throughout the iterative process, and have the role of reducing the variance introduced by compression. If no compression is used, we may simply set the shift terms to be $h_i^k = 0$ for all i, k.

Our method was inspired by Mishchenko et al. (2019), who first studied variance reduction for CGD methods for a specific ternary compression operator, and Horváth et al. (2019b) who studied the general class of ω-compression operators we also study here. However, we had to make certain modifications to make variance-reduced compression work in the accelerated case since both of them were studied in the non-accelerated case. Besides, our method adopts a randomized update rule for the auxiliary vectors w^k which simplifies the algorithm and analysis, resembling the workings of the loopless SVRG method proposed by Kovalev et al. (2020).

Algorithm 2 Accelerated DIANA (ADIANA)

```
Input: initial point $x^0$, $\{h_i^0\}_{i=1}^n$, $h^0 = \frac{1}{n}\sum_{i=1}^n h_i^0$, parameters $\eta, \theta_1, \theta_2, \alpha, \beta, \gamma, p$
1:  $z^0 = y^0 = w^0 = x^0$
2:  for $k = 0, 1, 2, \ldots$ do
3:      $x^k = \theta_1 z^k + \theta_2 w^k + (1-\theta_1-\theta_2) y^k$
4:      for all machines $i = 1, 2, \ldots, n$ do in parallel
5:          Compress shifted local gradient $\mathcal{C}_i^k(\nabla f_i(x^k) - h_i^k)$ and send to the server
6:          Update local shift $h_i^{k+1} = h_i^k + \alpha \mathcal{C}_i^k(\nabla f_i(w^k) - h_i^k)$
7:      end for
8:      Aggregate received compressed gradient information
            $g^k = \frac{1}{n}\sum_{i=1}^n \mathcal{C}_i^k(\nabla f_i(x^k) - h_i^k) + h^k$
            $h^{k+1} = h^k + \alpha \frac{1}{n}\sum_{i=1}^n \mathcal{C}_i^k(\nabla f_i(w^k) - h_i^k)$
9:      Perform update step
            $y^{k+1} = \mathrm{prox}_{\eta\psi}(x^k - \eta g^k)$
10:     $z^{k+1} = \beta z^k + (1-\beta) x^k + \frac{\gamma}{\eta}(y^{k+1} - x^k)$
11:     $w^{k+1} = \begin{cases} y^k, & \text{with probability } p \\ w^k, & \text{with probability } 1-p \end{cases}$
12: end for
```

5.2. Convergence theory

Our main convergence result for ADIANA (Algorithm 2) is formulated in Theorem 4. We focus on the strongly convex setting.

Theorem 4 *Suppose f is μ-strongly convex and that the functions f_i have L-Lipschitz continuous gradient for all i. Further, let the compression operator $\mathcal{C}$ satisfy (5). Choose the ADIANA (Algorithm 2) parameters as follows:*

$$\eta = \min\left\{\frac{1}{2L}, \frac{n}{64\omega(2p(\omega+1)+1)^2 L}\right\},$$
$$\theta_1 = \min\left\{\frac{1}{4}, \sqrt{\frac{\eta\mu}{p}}\right\}, \quad \theta_2 = \frac{1}{2},$$
$$\alpha = \frac{1}{\omega+1}, \quad \beta = 1-\gamma\mu, \quad \gamma = \frac{\eta}{2(\theta_1+\eta\mu)},$$
$$p = \min\left\{1, \frac{\max\{1, \sqrt{\frac{n}{32\omega}} - 1\}}{2(1+\omega)}\right\}.$$

Then the number of iterations performed by ADIANA to find an ϵ-solution such that

$$\mathbb{E}[\|z^k - x^*\|^2] \le \epsilon$$

is at most

$$k = \begin{cases} O\left(\left[\omega + \omega\sqrt{\frac{L}{n\mu}}\,\right]\log\frac{1}{\epsilon}\right), & n \le \omega, \\ O\left(\left[\omega + \sqrt{\frac{L}{\mu}} + \sqrt{\sqrt{\frac{\omega}{n}}\frac{\omega L}{\mu}}\,\right]\log\frac{1}{\epsilon}\right), & n > \omega. \end{cases}$$

As we have explained in the introduction, the above rate is vastly superior to that of non-accelerated distributed CGD methods, including that of DIANA.

5.3. Proof sketch

In the proof, we use the following notation:

$$\mathcal{Z}^k := \left\|z^k - x^*\right\|^2, \tag{6}$$
$$\mathcal{Y}^k := P(y^k) - P(x^*), \tag{7}$$
$$\mathcal{W}^k := P(w^k) - P(x^*), \tag{8}$$
$$\mathcal{H}^k := \frac{1}{n}\sum_{i=1}^n \left\|h_i^k - \nabla f_i(w^k)\right\|^2. \tag{9}$$

We first present a key technical lemma which plays a similar role to that of Lemma 1.

Lemma 2 *If the parameters satisfy $\eta \le \frac{1}{2L}, \theta_1 \le \frac{1}{4}, \theta_2 = \frac{1}{2}, \gamma = \frac{\eta}{2(\theta_1+\eta\mu)}$ and $\beta = 1-\gamma\mu$, then we have for any iteration k,*

$$\frac{2\gamma\beta}{\theta_1}\mathbb{E}\left[\mathcal{Y}^{k+1}\right] + \mathbb{E}\left[\mathcal{Z}^{k+1}\right]$$
$$\le (1-\theta_1-\theta_2)\frac{2\gamma\beta}{\theta_1}\mathcal{Y}^k + \beta\mathcal{Z}^k$$
$$+ 2\gamma\beta\frac{\theta_2}{\theta_1}\mathcal{W}^k + \frac{\gamma\eta}{\theta_1}\mathbb{E}\left[\left\|g^k - \nabla f(x^k)\right\|^2\right] \tag{10}$$
$$- \frac{\gamma}{4Ln\theta_1}\sum_{i=1}^n \left\|\nabla f_i(w^k) - \nabla f_i(x^k)\right\|^2 \tag{11}$$
$$- \frac{\gamma}{8Ln\theta_1}\sum_{i=1}^n \left\|\nabla f_i(y^k) - \nabla f_i(x^k)\right\|^2. \tag{12}$$

Theorem 4 can be proved by combing the above lemma with three additional Lemmas: Lemma 3, 4 and 5, which we present next. In view of the presence of $\mathcal{W}^k$ in (10), the following result is useful as it allows us to add $\mathcal{W}^{k+1}$ into the Lyapunov function.

Lemma 3 *According to Line 11 of Algorithm 2 and Definition (7)–(8), we have*

$$\mathbb{E}\left[\mathcal{W}^{k+1}\right] = (1-p)\mathcal{W}^k + p\mathcal{Y}^k.$$

To cancel the term $\mathbb{E}\left[\left\|g^k - \nabla f(x^k)\right\|^2\right]$ in (10), we use the defining property of compression operator (i.e., (5)) :

Lemma 4 *If the compression operator $\mathcal{C}$ satisfies (5), we have*

$$\mathbb{E}\left[\left\|g^k - \nabla f(x^k)\right\|^2\right]$$
$$\le \frac{2\omega}{n^2}\sum_{i=1}^n \left\|\nabla f_i(w^k) - \nabla f_i(x^k)\right\|^2 + \frac{2\omega}{n}\mathcal{H}^k. \tag{13}$$

Note that the bound on variance obtained above introduces an additional term $\mathcal{H}^k$ (see (13)). We will therefore add the terms $\mathcal{H}^{k+1}$ into the Lyapunov function as well.

Lemma 5 *If $\alpha \le \frac{1}{\omega+1}$, we have*

$$\begin{aligned}\mathbb{E}\left[\mathcal{H}^{k+1}\right] \le \left(1-\frac{\alpha}{2}\right)\mathcal{H}^k \\ + \left(1+\frac{2p}{\alpha}\right)\frac{2p}{n}\sum_{i=1}^n \left\|\nabla f_i(w^k)-\nabla f_i(x^k)\right\|^2 \\ + \left(1+\frac{2p}{\alpha}\right)\frac{2p}{n}\sum_{i=1}^n \left\|\nabla f_i(y^k)-\nabla f_i(x^k)\right\|^2.\end{aligned}$$

Note that the terms $\sum_{i=1}^n \left\|\nabla f_i(w^k)-\nabla f_i(x^k)\right\|^2$ and $\sum_{i=1}^n \left\|\nabla f_i(y^k)-\nabla f_i(x^k)\right\|^2$ in Lemma 5 and (13) can be cancelled by (11) and (12) by choosing the parameters appropriately.

Finally, it is not hard to obtain the following key inequality for the Lyapunov function by plugging Lemmas 3-5 into our key Lemma 2:

$$\begin{aligned}&\mathbb{E}\left[c_1\mathcal{Y}^{k+1}+c_2\mathcal{Z}^{k+1}+c_3\mathcal{W}^{k+1}+c_4\mathcal{H}^{k+1}\right] \\ &\le (1-c_5)\left(c_1\mathcal{Y}^k+c_2\mathcal{Z}^k+c_3\mathcal{W}^k+c_4\mathcal{H}^k\right). \qquad (14)\end{aligned}$$

Above, the constants $c_1,\ldots,c_5$ are related to the algorithm parameters $\eta,\theta_1,\theta_2,\alpha,\beta,\gamma$ and p. Finally, the proof of Theorem 4 can be derived (i.e., plug into the specified parameters) from inequality (14). The detailed proof can be found in the appendix.

6. Experiments

In this section, we demonstrate the performance of our accelerated method ADIANA (Algorithm 2) and previous methods with different compression operators on the regularized logistic regression problem,

$$\min_{x\in\mathbb{R}^d}\left\{\frac{1}{n}\sum_{i=1}^n \log\left(1+\exp(-b_i a_i^\top x)\right)+\frac{\lambda}{2}\|x\|^2\right\},$$

where $\{a_i,b_i\}_{i\in[n]}$ are data samples.

Data sets. In our experiments we use four standard datasets, namely, `a5a`, `mushrooms`, `a9a` and `w6a` from the LIBSVM library. Some of the experiments are provided in the appendix.

Compression operators. We use three different compression operators: random sparsification (see e.g. (Stich et al., 2018)), random dithering (see e.g. (Alistarh et al., 2017)), and natural compression (see e.g. (Horváth et al., 2019a)). For random-r sparsification, the number of communicated bits per iteration is $32r$, and we choose $r=d/4$. For random dithering, we choose $s=\sqrt{d}$, which means the number of communicated bits per iteration is $2.8d+32$ (Alistarh et al., 2017). For natural compression, the number of communicated bits per iteration is $9d$ bits (Horváth et al., 2019a).

Parameter setting. In our experiments, we use the theoretical stepsize and parameters for all the three algorithms: vanilla distributed compressed gradient descent (DCGD), DIANA (Mishchenko et al., 2019), and our ADIANA (Algorithm 2). The default number of nodes/machines is 20 and the regularization parameter $\lambda=10^{-3}$. The numerical results for different number of nodes can be found in the appendix. For the figures, we plot the relation of the optimality loss gap $f(x^k)-f(x^*)$ and the number of accumulated transmitted bits. The optimal value $f(x^*)$ for each case is obtained by getting the minimum of three uncompressed versions of ADIANA, DIANA, and DCGD for 100000 iterations.

6.1. Comparison with DIANA and DCGD

In this subsection, we compare our ADIANA with DIANA and DCGD with three compression operators: random sparsification, random dithering, and natural compression in Figures 1 and 2.

The experimental results indeed show that our ADIANA converges fastest for all three compressors, and natural compression uses the fewest communication bits than random dithering and random sparsification. Moreover, because the compression error of vanilla DCGD is nonzero in general, DCGD can only converge to the neighborhood of the optimal solution while DIANA and ADIANA can converge to the optimal solution.

6.2. Communication efficiency

Now, we compare our ADIANA and DIANA, with and without compression to show the communication efficiency of our accelerated method ADIANA in Figures 3 and 4.

According to the left top and left bottom of Figure 4, DIANA is better than its uncompressed version if the compression operator is random sparsification. However, ADIANA behaves worse than its uncompressed version. For random dithering (middle figures) and natural compression (right figures), ADIANA is about twice faster than its uncompressed version, and is much faster than DIANA with/without compression. These numerical results indicate that ADIANA (which enjoys both acceleration and compression) could be a more practical communication efficiency method, i.e., acceleration (better than non-accelerated DIANA) and compression (better than the uncompressed version), especially for random dithering and natural compression.

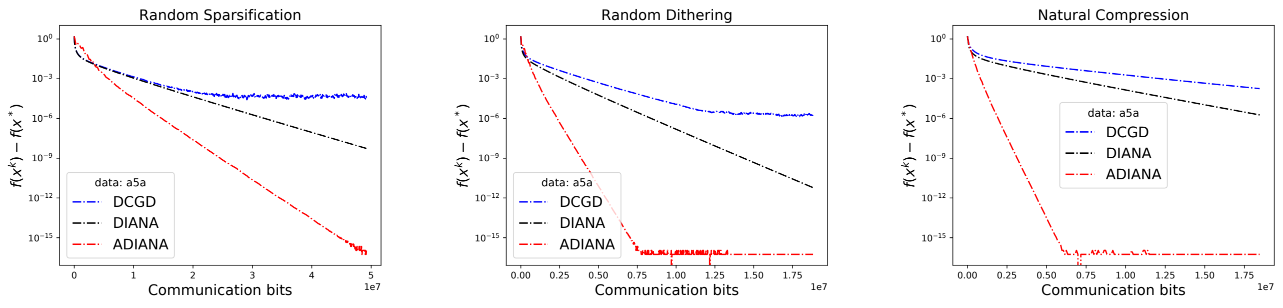

Figure 1. The communication complexity of different methods for three different compressors (random sparsification, random dithering and natural compression) on the `a5a` dataset.

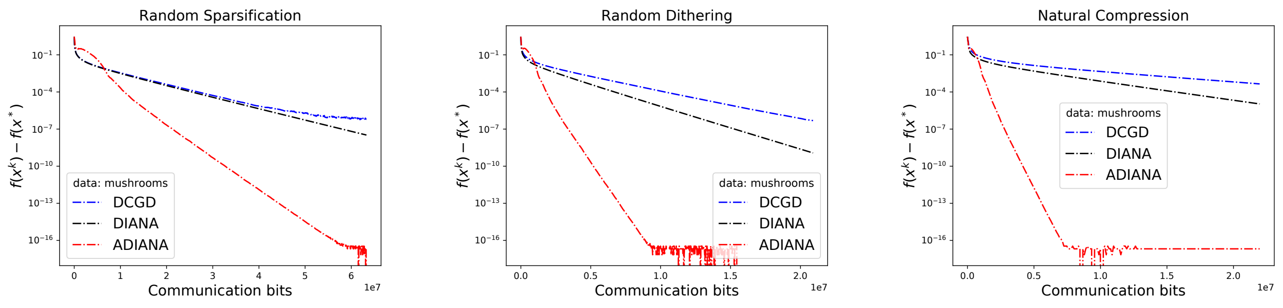

Figure 2. The communication complexity of different methods for three different compressors (random sparsification, random dithering and natural compression) on the `mushrooms` dataset.

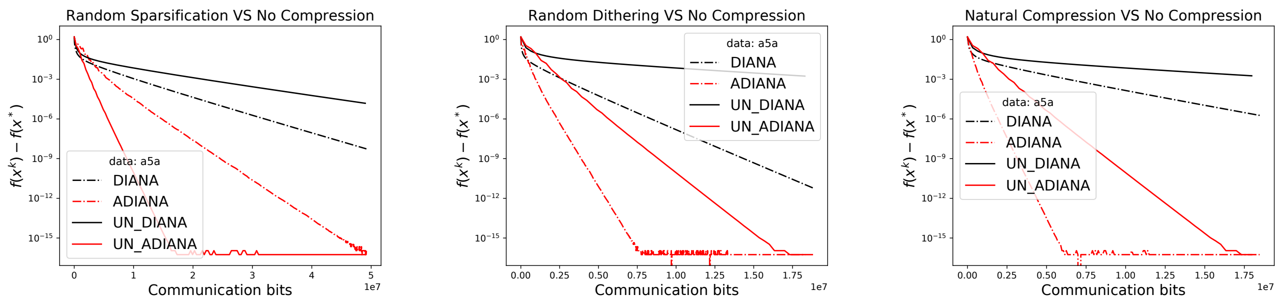

Figure 3. The communication complexity of DIANA and ADIANA with and without compression on the `a5a` dataset.

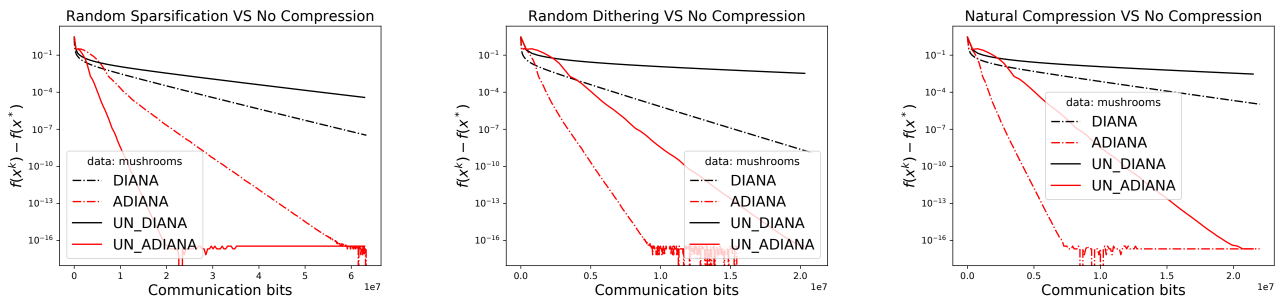

Figure 4. The communication complexity of DIANA and ADIANA with and without compression on the `mushrooms` dataset.

Acknowledgements

The authors would like to acknowledge support from the KAUST Baseline Research Fund and the KAUST Visual Computing Center. Zhize Li and Xun Qian thank for support from the KAUST Extreme Computing Research Center.

References

Alistarh, D., Grubic, D., Li, J., Tomioka, R., and Vojnovic, M. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pp. 1709–1720, 2017.

Allen-Zhu, Z. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1200–1205. ACM, 2017.

Allen-Zhu, Z., Qu, Z., Richtárik, P., and Yuan, Y. Even faster accelerated coordinate descent using non-uniform sampling. In *The 33rd International Conference on Machine Learning*, pp. 1110–1119, 2016.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Hanzely, F. and Richtárik, P. Accelerated coordinate descent with arbitrary sampling and best rates for minibatches. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 2019a.

Hanzely, F. and Richtárik, P. One method to rule them all: variance reduction for data, parameters and many new methods. *arXiv preprint arXiv:1905.11266*, 2019b.

Hanzely, F., Mishchenko, K., and Richtárik, P. SEGA: variance reduction via gradient sketching. In *Advances in Neural Information Processing Systems 31*, pp. 2082–2093, 2018.

Horváth, S., Ho, C.-Y., Ľudovít Horváth, Sahu, A. N., Canini, M., and Richtárik, P. Natural compression for distributed deep learning. *arXiv preprint arXiv:1905.10988*, 2019a.

Horváth, S., Kovalev, D., Mishchenko, K., Stich, S., and Richtárik, P. Stochastic distributed learning with gradient quantization and variance reduction. *arXiv preprint arXiv:1904.05115*, 2019b.

Karimireddy, S., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. SCAFFOLD: Stochastic controlled averaging for on-device federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

Khaled, A., Mishchenko, K., and Richtárik, P. First analysis of local GD on heterogeneous data. In *NeurIPS Workshop on Federated Learning for Data Privacy and Confidentiality*, pp. 1–11, 2019.

Khaled, A., Mishchenko, K., and Richtárik, P. Tighter theory for local SGD on identical and heterogeneous data. In *The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS 2020)*, 2020.

Khirirat, S., Feyzmahdavian, H. R., and Johansson, M. Distributed learning with compressed gradients. *arXiv preprint arXiv:1806.06573*, 2018.

Kingma, D. P. and Ba, J. Adam: a method for stochastic optimization. In *The 3rd International Conference on Learning Representations*, 2014.

Konečný, J., McMahan, H. B., Yu, F., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: strategies for improving communication efficiency. In *NIPS Private Multi-Party Machine Learning Workshop*, 2016.

Kovalev, D., Horváth, S., and Richtárik, P. Don't jump through hoops and remove those loops: SVRG and Katyusha are better without the outer loop. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, 2020.

Lan, G., Li, Z., and Zhou, Y. A unified variance-reduced accelerated gradient method for convex optimization. In *Advances in Neural Information Processing Systems*, pp. 10462–10472, 2019.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: challenges, methods, and future directions. *arXiv preprint arXiv:1908.07873*, 2019.

Li, Z. and Richtárik, P. A unified analysis of stochastic gradient methods for nonconvex federated optimization. *arXiv preprint arXiv:2006.07013*, 2020.

Ma, C., Konečný, J., Jaggi, M., Smith, V., Jordan, M. I., Richtárik, P., and Takáč, M. Distributed optimization with arbitrary local solvers. *Optimization Methods and Software*, 32(4):813–848, 2017.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Agüera y Arcas, B. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

Mishchenko, K., Gorbunov, E., Takáč, M., and Richtárik, P. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

Nesterov, Y. A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$. In *Doklady AN USSR*, volume 269, pp. 543–547, 1983.

Nesterov, Y. *Introductory lectures on convex optimization: a basic course*. Kluwer Academic Publishers, 2004.

Nesterov, Y. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Qian, X., Qu, Z., and Richtárik, P. L-SVRG and L-Katyusha with arbitrary sampling. *arXiv preprint arXiv:1906.01481*, 2019.

Seide, F., Fu, H., Droppo, J., Li, G., and Yu, D. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

Stich, S. U. Local SGD converges fast and communicates little. In *International Conference on Learning Representations*, 2019.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems*, pp. 4447–4458, 2018.

Wangni, J., Wang, J., Liu, J., and Zhang, T. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, pp. 1306–1316, 2018.